On the Equivalence of Strong Formulations for Capacitated Multi-level Lot Sizing Problems with Setup Times

Tao Wu[1], Leyuan Shi[1], Joseph Geunes[2], Kerem Akartunalı[3]

[1] *Department of Industrial and Systems Engineering, University of Wisconsin-Madison, Madison, WI 53706*

[2] *Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611*

[3] *Department of Management Science, University of Strathclyde, Glasgow G1 1QE, United Kingdom*

Abstract

Several mixed integer programming formulations have been proposed for modeling capacitated multi-level lot sizing problems with setup times. These formulations include the so-called Facility Location formulation, the Shortest Route formulation, and the Inventory and Lot Sizing formulation with (ℓ, S) inequalities. In this paper, we demonstrate the equivalence of these formulations when the integrality requirement is relaxed for any subset of binary setup decision variables. This equivalence has significant implications for decomposition-based methods since same optimal solution values are obtained no matter which formulation is used. In particular, we discuss the Relax-and-Fix method, a decomposition-based heuristic used for the efficient solution of hard lot sizing problems. Computational tests allow us to compare the effectiveness of different formulations using benchmark problems. The choice of formulation directly affects the required computational effort, and our results therefore provide guidelines on choosing an effective formulation during the development of heuristic-based solution procedures.

Key words: Capacitated Multi-level Lot Sizing, Inventory and Lot Sizing, Facility Location, Shortest Route, Relax-and-Fix.

1. Introduction

The Capacitated Multi-level Lot Sizing Problem with Setup Times (CLST-ML) has received a great deal of attention in past literature on operations and optimization. Effective solutions for this problem class play a significant role in the effective utilization of limited resources in many practical production systems. Various Mixed Integer Programming (MIP) formulations have been proposed for modeling the CLST-ML in the literature. The three most prevalent of these formulations are the Inventory and Lot Sizing (ILS) formulation, the Facility Location (FL) formulation, and the Shortest Route (SR) formulation, originally proposed by Billington *et al.* [1983], Krarup and Bilde [1977], and Eppen and Martin [1987], respectively. Subsequent research has demonstrated important relationships among these formulations, as well as the advantages and disadvantages of each of these formulations. To the best of our knowledge, the first of these equivalence results was shown by Nemhauser and Wolsey [1988], who showed the equivalence of the FL and SR formulations for the Uncapacitated Lot Sizing Problem (ULSP) by showing that the Linear Programming (LP) relaxations of both formulations are tight (in the sense that the optimal value of the LP relaxation equals that of the original MIP). Based on these results, they also demonstrated the equivalence of the LP relaxations of these two formulations in the presence of additional complicating constraints. Denizel *et al.* [2008] subsequently showed the equivalence of the LP relaxations of the FL and SR formulations for the CLST-ML. Akartunalı and Miller [2009] added (l, S) separation cuts to the ILS formulation, and then showed that this strengthened version of the formulation provides the same LP lower bounds as the FL and SR formulations for the CLST-ML. Furthermore, Akartunalı and Miller [2007] compare a wide variety of different lower bounds for the CLST-ML.

The existing equivalence results in the literature provide assistance and insight into the best choice of formulation for generating lower bounds on the optimal solution value. However, a number of exact and heuristic solution methods require the solution of smaller subproblems in which the integrality requirement is relaxed for a subset of the binary setup decision variables (such that the associated relaxed variables may take any value between 0 and 1 inclusive). Obviously, the existing equivalence results in the literature do not directly apply to these subproblems. The objective of this paper is to provide theoretical results to demonstrate that the ILS with (l, S) inequalities, FL, and SR formulations remain equivalent for any such subproblems.

The equivalence of these subproblems implies that many prominent decomposition-based heuristic methods in the lot sizing literature, such as Relax-and-Fix (a time-oriented decomposition method), will lead to a solution with the same objective function value, no matter which of these three formulations is utilized. This equivalence may also have useful implications for the choice of formulation to use within a customized branch-and-bound algorithm. Clearly, if two formulations provide equivalent lower bounds on objective function value for a minimization problem (when solving any subproblem in which a common subset of binary variables is fixed and the remaining binary variables are relaxed), we should choose the one whose LP relaxation can be solved the most quickly in order to facilitate fast exploration of the branch and bound tree. If a feasible solution (upper bound) is also generated at each node via some heuristic, and if such a heuristic uses an LP relaxation as its starting point (as in, e.g., an LP rounding approach or a Relax-and-Fix approach), then the formulation that produces the heuristic solution the fastest should be selected (assuming the different formulations lead to the same heuristic solution value, as we later show occurs for the Relax-and-Fix heuristic). While solving separate relaxations may duplicate some effort, a customized parallel implementation may enable overcoming this limitation.

The equivalence of these three formulations might also have useful implications for a branch-and-cut method. This is because a cutting plane generated for one formulation might be directly projected to the place of another formulation. The equivalency continues to hold for the two formulations with cutting planes, and then the above procedure is still applicable. For example, a cutting plane for the S-ILS formulation can be exactly expressed for the FL and SR formulations. In other words, no matter what cutting planes have been generated using the S-ILS formulation, a corresponding equivalent formulation for FL and SR can be easily found using the relationships of decision variables in these formulations.

In this paper, we will use the Relax-and-Fix heuristic method as an example heuristic approach that permits computationally demonstrating the equivalence of the three formulations. The Relax-and-Fix heuristic is a priority-oriented method in general, and a time-oriented decomposition method in the case of lot sizing. It is a relatively simple and straightforward approach for decomposing the original large-scale Mixed Integer Linear Programming (MILP) lot-sizing problem into several smaller MILP problems by restricting the binary variables in certain time periods, and relaxing the binary variables so they are continuous in the remaining periods. In the early stages of such an algorithmic approach, only the binary variables in the first "time windows" are treated as binary variables, and all other binary variables are relaxed so as to be continuous. These "time windows" correspond to time intervals spanning several contiguous periods (e.g., a time interval from period 1 to period 4 is a time window). This permits solving a smaller problem (with fewer "complicating" binary variables) using an MILP solver, and using the resulting solution to fix the binary variables within the window. The next time window is then processed in the same manner, and so on, until the last time window is completed. The Relax-and-Fix heuristic has been implemented in a number of past works, including Belvaux and Wolsey [2000], Stadtler [2003], Mercé and Fontan [2003], Absi and Kedad-Sidhoum [2007], Akartunalı and Miller [2009], Sahling *et al.* [2009], and Wu *et al.* [2010], to name a few.

Because of the equivalent solution quality produced by these formulations, computational time becomes a key factor in choosing a problem formulation. To characterize the efficiency of the formulations in generating lower and upper bounds, we will compare the resulting computational performance in generating heuristic solutions using Relax-and-Fix. Our experimental tests were conducted on the data sets provided by Stadtler [2003] in order to provide guidance on the choice among the ILS with (ℓ, S) inequalities, FL, and SR formulations. These computational results show that the preference order of formulations consists of ILS with (ℓ, S) inequalities, FL, and then SR, when a Relax-and-Fix algorithm is implemented. The theoretical and empirical results presented in this paper play a significant role in model formulation choice, and provide direction on the development of heuristic-based solution procedures.

The remainder of this paper is organized as follows: In Section 2, we provide three prominent model formulations for the CLST-ML and discuss the strengths and weaknesses of each modeling approach. In Section 3, we provide results on the equivalence of the ILS with (ℓ, S) inequalities, FL, and SR formulations for the continuous relaxation in which some common subset of binary setup decision variables is permitted to take continuous values (and the remaining setup variables must take binary values). In Section 4, equivalence results pertaining to heuristic methods are analyzed, with the Relax-and-Fix method used as an example. In Section 5, we present computational results obtained using a number of previously published data sets. Finally, we conclude with future directions in Section 6.

2. Problem Model Formulations

In this section we present three well-known formulations for CLST-ML. The following assumptions are employed in our model. Setup times and costs are non-sequence dependent, the possibility of setup carryover between periods is neglected, and shortages and non-zero initial inventories are not permitted. All other production costs are assumed to be time-invariant and linear in production output; therefore they are omitted. Note that this is for formulation simplicity and our results remain valid as long as the production costs remain linear. Setup costs and holding costs are also assumed to be time-invariant; this assumption is for the sake of expositional simplicity, and the theoretical results in this paper would be also valid in the case of time-varying setup and holding costs. We also assume that end items do not have any successors, and only end items have external demands. We first present the notation we will use before providing specific problem formulations.

Indices and index sets:

T	number of time periods in the planning horizon.
M	number of production resources or machines.
I	number of items (subassemblies and/or end items).
I^E	number of end items.
q, p, t	interchangeable time period indices, $q, p, t \in \{1, \ldots, T\}$.
m	index for machines, m=1,. . . ,M.
i, j	item indices, $i, j \in \{1, \ldots, I\}$. If $i, j \in \{1, \ldots, I^E\}$, then this corresponds to an end item. If $i, j \in \{I^E + 1, \ldots, I\}$, then this corresponds to an non-end item.
η_i	set of immediate successors of item i.

Parameters:

sc_i	setup cost for producing a lot of item i.
hc_i	inventory holding cost for one unit of item i remaining at the end of a period.
ec_i	echelon inventory holding cost for one unit of item i remaining at the end of a period.
st_{im}	setup time required for producing item i on machine m.
a_{im}	production time required to produce a unit of item i on machine m.
gd_{it}	gross demand for item i in period t.
gd_{itp}	total gross demand for item i from period t to period p.
ed_{it}	echelon demand for item i in period t.
ed_{itp}	total echelon demand for item i from period t to period p.
r_{ij}	number of units of item i needed to produce one unit of the immediate successor item j, and $r_{ij} = 0$ when $i = j$
C_{mt}	available capacity of machine m in period t.
BM_{it}	maximum number of units of item i that can be produced in period t.

Variables:

x_{it}	number of units of item i produced in period t.
s_{it}	inventory of item i at the end of period t.
e_{it}	echelon inventory of item i at the end of period t.
y_{it}	binary setup decision variables ($y_{it} = 1$ if production is setup for item i in period t, and 0 otherwise).
u_{itp}	number of units of item i produced in period t to satisfy demand in period p ($t \leq p$).
w_{itp}	percentage of production of item i in period t ($t \leq p$) used to satisfy accumulated demand for item i from period t to period p.

2.1. The Inventory and Lot Sizing Formulation

Several model formulations yielding different formulation sizes and relaxation lower bounds have been proposed for CLST-ML. We start with the model formulation of Billington *et al.* [1983], which is referred to as the ILS formulation because of the use of explicit inventory variables. The formulation for the CLST-ML problem is given as follows.

ILS:

$$\min \sum_{i=1}^{I} \sum_{t=1}^{T} sc_i \cdot y_{it} + \sum_{i=1}^{I} \sum_{t=1}^{T} hc_i \cdot s_{it} \tag{1}$$

Subject to:

$$x_{it} + s_{i(t-1)} = gd_{it} + s_{it} \qquad \forall\, i = 1, \ldots, I^E, t = 1, \ldots, T, \tag{2}$$

$$x_{it} + s_{i(t-1)} = gd_{it} + \sum_{j \in \eta_i} r_{ij} \cdot x_{jt} + s_{it} \qquad \forall\, i = I^E + 1, \ldots, I, t = 1, \ldots, T, \tag{3}$$

$$\textstyle\sum_{i=1}^{I} a_{im} \cdot x_{it} + \sum_{i=1}^{I} st_{im} \cdot y_{it} \leq C_{mt} \qquad \forall\, m = 1, \ldots, M, t = 1, \ldots, T, \tag{4}$$

$$x_{it} \leq BM_{it} \cdot y_{it} \qquad \forall\, i = 1, \ldots, I, t = 1, \ldots, T, \tag{5}$$

$$x_{it} \geq 0,\ s_{it} \geq 0,\ s_{i0} = 0,\ y_{it} \in \{0, 1\} \qquad \forall\, i = 1, \ldots, I, t = 1, \ldots, T. \tag{6}$$

The objective function minimizes total setup and holding costs during the planning horizon. Constraints (2) and (3) ensure demand satisfaction in all periods for end- and non-end items, respectively. Constraints (4) enforce capacity requirements. Constraint set (5) ensures that no production occurs for item i in period t unless the corresponding binary setup variable, y_{it}, takes a value of 1, in which case the amount of production is limited only by the value of BM_{it}. Here, BM_{it} can be defined formally as $BM_{it} = \min\left(\sum_{j \in \eta_i} r_{ij} \cdot gd_{jtT}, \frac{C_{mt} - st_{im}}{a_{im}}\right)$, for $i = 1, \ldots, I$ and $t = 1, \ldots, T$. The BM_{it} values are defined in the same way for each of the following formulations. Constraints (6) enforce the binary and nonnegative requirements for different variables.

The model size corresponding to the ILS formulation is relatively modest, implying that its LP relaxation is readily solvable for problem instances with a few hundred items over a large planning interval. However, the time required for proving the optimality of a given solution is often prohibitive because the integrality gap associated with the LP relaxation is typically very large. Furthermore, the poor lower bound provided by the LP relaxation usually will not be adequate to guide the search for good feasible solutions in the branch-and-bound phase of a standard MIP solver. Consequently, strong inequalities need to be added to the model to provide a tighter lower bound. Among the inequalities proposed in previous research, the (ℓ, S) inequalities proposed by Barany *et al.* [1984] are very efficient in improving the lower bound. By defining echelon demand parameters ed_{it} and echelon stock variables e_{it}, where $ed_{it} = gd_{it} + \sum_{j\in\eta_i} r_{ij} \cdot ed_{jt}$ and $e_{it} = s_{it} + \sum_{j\in\eta_i} r_{ij} \cdot e_{jt}$ for $t = 1, \ldots, T$ and $i = I^E + 1, \ldots I$, respectively, and $ed_{it} = gd_{it}$ and $e_{it} = s_{it}$ for $t = 1, \ldots, T$ and $i = 1, \ldots, I^E$, respectively, the (ℓ, S) inequalities can be written as

$$\sum_{t\in S} x_{it} \leq \sum_{t\in S} ed_{it\ell} \cdot y_{it} + e_{i\ell}. \quad \forall\, i = 1, \ldots, I, \ell = 1, \ldots, T, S \subseteq [1, \ell]. \tag{7}$$

We refer to the ILS formulation with the (ℓ, S) inequalities as S-ILS throughout the rest of the paper. Although the number of (ℓ, S) inequalities increases exponentially in T, a polynomial-time separation algorithm was provided by Barany *et al.* [1984]. The (ℓ, S) separation algorithm is stated as follows. In the algorithm, a prime symbol $(')$ denotes the value of the corresponding variable in the solution; for example, x'_{it} denotes the value of x_{it} in the problem solution.

Algorithm 1: The (ℓ, S) Separation Algorithm **(Barany *et al.* [1984])**

```
Set $noviolation = 0$ ;
repeat
    Solve the LP relaxation of ILS, and obtain the solution values $x'_{it}$, $y'_{it}$, and $e'_{it}$
    ($\forall\, i = 1, \ldots, I,\ t = 1, \ldots, T$) ;
    for $i = 1, ..., I$ do
        for $\ell = 1, ..., T$ do
            Initialize $S = \emptyset$ ;
            for $t = 1, ..., \ell$ do
                if $x'_{it} > ed_{it,\ell} \cdot y'_{it}$ then
                    $S = S \cup t$ ;
            if $x'_{it} > ed_{it,\ell} \cdot y'_{it}$ then
                Set $noviolation = noviolation$ +1 and add the violated $(\ell,\ S)$
                inequalities into ILS ;
until $noviolation \neq 0$ ;
```

2.2. *The Facility Location Formulation*

The FL formulation was originally proposed for single-item problems by Krarup and Bilde [1977] and is given as follows.

FL:

$$\min \sum_{i=1}^{I}\sum_{t=1}^{T} sc_i \cdot y_{it} + \sum_{i=1}^{I}\sum_{t=1}^{T}\sum_{p=t}^{T} ec_i \cdot (p-t) \cdot u_{itp} \tag{8}$$

Subject to:

$$\sum_{t=1}^{p} u_{itp} = ed_{ip} \qquad \forall\, i = 1,\ldots,I, p = 1,\ldots,T, \tag{9}$$

$$s_{i,t-1} + \sum_{p=t}^{T} u_{itp} = gd_{it} + \sum_{j\in\eta_i}\sum_{p=t}^{T} r_{ij}u_{jtp} + s_{it} \qquad \forall\, i = I^E+1,\ldots,I,\; t = 1,\ldots,T, \tag{10}$$

$$u_{itp} \le ed_{ip}y_{it} \qquad \forall\, i = 1,\ldots,I, t = 1,\ldots,T,\; p = t,\ldots,T. \tag{11}$$

$$\sum_{i=1}^{I}\sum_{p=t}^{T} a_{im}u_{itp} + \sum_{i=1}^{I} st_{im}y_{it} \le C_{mt} \qquad \forall\, m = 1,\ldots,M,\; t = 1,\ldots,T, \tag{12}$$

$$\sum_{p=t}^{T} u_{itp} \le BM_{it}y_{it} \qquad \forall\, i = 1,\ldots,I, t = 1,\ldots,T, \tag{13}$$

$$u_{itp} \ge 0,\; s_{it} \ge 0,\; s_{i0} = 0,\; y_{it} \in \{0,1\} \qquad \forall\, i = 1,\ldots,I, t = 1,\ldots,T,\; p = t,\ldots,T. \tag{14}$$

Here, Constraints (9) ensure demand satisfaction in all periods for all items. Constraints (10) make sure that all predecessor items are available in the period of production for a corresponding production lot. Constraints (11) correspond to setup forcing constraints, while Constraint set (12) enforces production capacity limits. Constraints (13) ensure that the total amount of production is less than or equal to a sufficiently large value, and that a setup is performed in period t for item i if any demand is satisfied using the corresponding production. Observe that either of the constraint sets (11) and (13) may be dropped from the MILP formulation, and the remaining formulation provides a correct statement of the problem. However, the LP relaxation lower bound of the formulation is stronger if both of the constraints are included. Constraints (14) enforce the binary and nonnegativity requirements for different variables. We note that this FL formulation (as well as the SR formulation in the following subsection) is often presented in the literature without explicit inventory (s_{it}) variables, as these may be substituted out of the problem.

2.3. The Shortest Route Formulation

Improved model formulations based on the SR representation have been proposed by Eppen and Martin [1987] for the single-level case and extended by Tempelmeier and Helber [1994] to the multi-level case. The attractiveness of the SR model stems from the fact that its LP relaxation yields strong lower bounds. The SR formulation is given as follows.

SR:

$$\min \sum_{i=1}^{I}\sum_{t=1}^{T}\sum_{p=t}^{T}\left\{\sum_{q=t}^{p} ec_i \cdot (q-t) \cdot ed_{iq}\right\} \cdot w_{itp} + \sum_{i=1}^{I}\sum_{t=1}^{T} sc_i \cdot y_{it} \tag{15}$$

Subject to:

$$\sum_{p=1}^{T} w_{i1p} = 1 \qquad \forall\, i = 1, ..., I, \tag{16}$$

$$\sum_{q=1}^{t-1} w_{iq(t-1)} = \sum_{p=t}^{T} w_{itp} \qquad \forall\, i = 1, ..., I, t = 2, ..., T, \tag{17}$$

$$\sum_{q=1}^{t}\sum_{p=q}^{T} ed_{iqp} \cdot w_{iqp} = \sum_{q=1}^{t} gd_{iq} + \sum_{j\in\eta_i}\sum_{q=1}^{t}\sum_{p=q}^{T} r_{ij} \cdot ed_{jqp} \cdot w_{jqp} + s_{it} \qquad \forall\, t = 1, ..., T, i = I^E + 1, ..., I, \tag{18}$$

$$\sum_{p=t}^{T} w_{itp} \leq y_{it} \qquad \forall\, i = 1, ..., I, t = 1, ..., T, \tag{19}$$

$$\sum_{i=1}^{I}\sum_{p=t}^{T} a_{im} \cdot ed_{itp} \cdot w_{itp} + \sum_{i=1}^{I} st_{im} \cdot y_{it} \leq C_{mt} \quad \forall\, m = 1, \ldots, M, t = 1, \ldots, T, \tag{20}$$

$$\sum_{p=t}^{T}\sum_{q=p}^{T} w_{itq} \cdot ed_{ip} \leq BM_{it} \cdot y_{it} \qquad \forall\, i = 1, \ldots, I, t = 1, \ldots, T, \tag{21}$$

$$w_{itp} \geq 0,\ s_{it} \geq 0,\ y_{it} \in \{0, 1\} \qquad \forall\, i = 1, \ldots, I, t = 1, \ldots, T. \tag{22}$$

Here, Constraints (16) and (17) ensure demand satisfaction in every period for all items. Constraints (18) make sure that all predecessor items are available in the period of production of a corresponding production lot. Constraints (19) are setup forcing constraints, and Constraints (20) enforce capacity limits. As in the previous FL formulation, Constraint set (21) is not required for a correct MILP formulation of the problem, but serves to improve the quality of the LP relaxation. This constraint set ensures that total production does not exceed some big value if a setup is performed, and that no corresponding production occurs if the associated binary setup variable equals zero. Constraints (22) enforce the binary and nonnegativity requirements for different variables.

3. Equivalence of Strong Formulations

This section presents equivalence results for the various formulations we have discussed, as well as for certain variations of these formulations. Akartunalı and Miller [2009] proved that the LP relaxations of S-ILS, FL, and SR yield the same lower bound for the CLST-ML problem. To the best of our knowledge, no other theoretical results have been demonstrated on their relationships. However, in our preliminary experiments, when we compared the results of the Relax-and-Fix algorithm using either S-ILS, FL, or SR, we found that the heuristic results were independent of the formulation used. A closer examination revealed that it was not a coincidence that the different formulations yielded the same heuristic solutions. This is because the formulations are equivalent when any common subset of the binary setup decision variables (y_{it}) is relaxed to be continuous.

Before presenting the main results of this section, we introduce some notation that will help in our exposition. First we define the following two constraints in order to be able to state the FL and SR formulations in spaces that include the production variable (x) dimension:

$$x_{it} = \sum_{p=t}^{T} u_{itp} \qquad \forall\, t = 1,\ldots,T, i = 1,\ldots,I. \tag{23}$$

$$x_{it} = \sum_{p=t}^{T} ed_{itp} w_{itp} \qquad \forall\, t = 1,\ldots,T, i = 1,\ldots,I. \tag{24}$$

Next, we define the feasible region associated with each formulation and the corresponding decision problem as follows:

$$\begin{aligned} &P_{S-ILS} = \{(x,y,s)|(2)-(7)\} && Z_{S-ILS} = \min\{(1)|(x,y,s) \in P_{S-ILS}\} \\ &P_{FL} = \{(x,y,s,u)|(9)-(14),(23)\} && Z_{FL} = \min\{(8)|(x,y,s,u) \in P_{FL}\} \\ &P_{SR} = \{(x,y,s,w)|(16)-(22),(24)\} && Z_{SR} = \min\{(15)|(x,y,s,w) \in P_{SR}\} \end{aligned}$$

Note that we will use the superscript LP to indicate the LP relaxation of a feasible region or problem, and the superscript SUB to indicate a feasible region for a subproblem (or the subproblem itself) in which only a subset of setup decision binary variables is relaxed to be continuous. That is, P_{S-ILS}^{LP} refers to the feasible region of S-ILS when the binary variables are relaxed, and P_{FL}^{SUB} refers to the feasible region of a subproblem of FL in which a subset of the binary variables is restricted to be binary, and the remaining binary variables are relaxed to be continuous. For a clearer presentation, we additionally employ the following definitions.

Definition 1: *Let* $\mathbb{A} \subseteq \{(x,z) \in \mathbb{R}^n \times \mathbb{R}^m\}$*; the projection of* $\mathbb{A}$ *onto the space* $x \in \mathbb{R}^n$*, denoted by* $proj_x(\mathbb{A})$*, is the set* $\{x \in \mathbb{R}^n : \exists z \in \mathbb{R}^m, (x,z) \in \mathbb{A}\}$.

In this definition, $\mathbb{A}$ is a subset of the Euclidean space, including polyhedra and mixed-integer sets, and $\mathbb{A}$ can be any of the previously defined feasible regions, e.g., $\mathbb{A}$ may correspond to P_{FL}. In addition, the projection of P_{FL} onto the space of the SR formulation is denoted by $proj_{x,y,s,w}(P_{FL})$.

As mentioned earlier, we have the following result from the literature.

Proposition 1. (Akartunalı and Miller [2009]) $Z_{S-ILS}^{LP} = Z_{FL}^{LP} = Z_{SR}^{LP}$.

Next, we consider the special case in which a subset $\mathcal{R} \subseteq I \times T$ exists for which the corresponding y variables are fixed to specific binary values, and we propose the following.

Proposition 2. *When a subset* $\mathcal{R}$ *of* y *variables is fixed,* $Z_{S-ILS}^{LP} = Z_{FL}^{LP}$ *remains valid.*

Proof of Proposition 2. Let $\mathcal{R} = \mathcal{R}_0 \cup \mathcal{R}_1$, where the first subset corresponds to y variables that are fixed to 0, and the second subset corresponds to y variables that are fixed to 1. We define new decision problems in this case as follows:

$$Z'_{S-ILS} = \min\{(1)|(x,y,s) \in P^{LP}_{S-ILS} \cap \{y_{it} \leq 0\ \forall i,t \in \mathcal{R}_0,\ \ y_{it} \geq 1\ \forall i,t \in \mathcal{R}_1\}\}$$
$$Z'_{FL} = \min\{(8)|(x,y,s,u) \in P^{LP}_{FL} \cap \{y_{it} \leq 0\ \forall i,t \in \mathcal{R}_0,\ \ y_{it} \geq 1\ \forall i,t \in \mathcal{R}_1\}\}$$

Next we define, for each of the above problems, the Lagrangean relaxation in which the the constraints that fix the y variables are relaxed (with corresponding Lagrange multipliers denoted λ), as follows:

$$LR_{S-ILS}(\lambda) = \min\{(1) + \sum_{i,t \in \mathcal{R}_0} \lambda_{it} y_{it} + \sum_{i,t \in \mathcal{R}_1} \lambda_{it}(1 - y_{it})|(x,y,s) \in P^{LP}_{S-ILS}\}$$
$$LR_{FL}(\lambda) = \min\{(8) + \sum_{i,t \in \mathcal{R}_0} \lambda_{it} y_{it} + \sum_{i,t \in \mathcal{R}_1} \lambda_{it}(1 - y_{it})|(x,y,s,u) \in P^{LP}_{FL}\}$$

Observe that for a given vector λ, $LR_{S-ILS}(\lambda) = LR_{FL}(\lambda)$, which simply follows from Proposition 1. The two equations below follow from Lagrangian duality:

$$\max_{\lambda \geq 0} LR_{S-ILS}(\lambda) = Z'_{S-ILS}$$
$$\max_{\lambda \geq 0} LR_{FL}(\lambda) = Z'_{FL}$$

Since $LR_{S-ILS}(\lambda) = LR_{FL}(\lambda)$ for a given λ, this implies $Z'_{S-ILS} = Z'_{FL}$, which concludes the proof.

Next, we generalize this result to problems in which a subset of y variables is required to take binary values (and the remaining y variables may be continuous), and we present our main result.

Theorem 1. $Z^{SUB}_{S-ILS} = Z^{SUB}_{FL}$.

Proof of Theorem 1. We define the following decision problems, where $\mathcal{R}$ corresponds to some subset of the y variables:

$$Z''_{S-ILS} = \min\{(1)|(x,y,s) \in P^{LP}_{S-ILS} \cap \{y_{it} \in \{0,1\}\ \forall i,t \in \mathcal{R}\}\}$$
$$Z''_{FL} = \min\{(8)|(x,y,s,u) \in P^{LP}_{FL} \cap \{y_{it} \in \{0,1\}\ \forall i,t \in \mathcal{R}\}\}$$

Note that for either of these problems, the binary search tree is exactly the same, i.e., each tree contains the same number of nodes, and each node in a given tree has a corresponding node in the other tree in which the fixed y variables are the same. Consider any node such that all variables in $\mathcal{R}$ must take values of either 0 or 1. Hence, at any such node we let $\mathcal{R} = \mathcal{R}_0 \cup \mathcal{R}_1$, where the first subset indicates y variables taking a value of 0, and the second subset indicates y variables taking a value of 1. We know from Proposition 2 that these problems are equivalent in either binary search tree, and, therefore, each node of the binary search tree has a corresponding equivalent node in the other search tree. Hence, we can conclude that $Z^{SUB}_{S-ILS} = Z^{SUB}_{FL}$.

Corollary 1. $Z^{SUB}_{S-ILS} = Z^{SUB}_{FL} = Z^{SUB}_{SR}$

The proof of this corollary follows exactly the same logic as the preceding proposition and theorem and we therefore omit the details.

As having equivalent objective function values for the three formulations does not necessarily imply the equivalence of the feasible sets, we next provide two results that show the relationship between the feasible sets of the formulations (S-ILS, FL, and SR) when the integrality requirement is relaxed for any subset of binary setup decision variables.

Theorem 2. $proj_{(x,y,s,e)}\left(P^{SUB}_{FL}\right) \subseteq P^{SUB}_{S-ILS}$.

In other words, the projection of P^{SUB}_{FL} onto the space of S-ILS is a subset of P^{SUB}_{S-ILS}, but we do not necessarily have this relationship holding in the opposite direction. We refer the interested reader to Appendix A for the proof.

Theorem 3. $proj_{(x,y,s,w)}\left(P^{SUB}_{FL}\right) = P^{SUB}_{SR}$.

In other words, the projection of P^{SUB}_{FL} onto the space of SR is equivalent to P^{SUB}_{SR}. We refer the reader to Appendix B for the proof.

4. Equivalence of Strong Formulations for Relax-and-Fix

In this section, we discuss the equivalence of the strong formulations under the Relax-and-Fix heuristic method. We first provide an illustrative example in order to describe the method clearly. Consider a 12-period problem with 40 items, which implies a total of 480 binary setup decision variables. Problems involving such a large number of binary variables are typically difficult to solve optimally, and it is typically difficult to even obtain a good feasible solution in acceptable time using an MILP solver. In order to reduce the difficulty and generate a good solution value, the Relax-and-Fix approach only restricts a subset of the binary decision variables to take values of 0 or 1, and relaxes the remaining subset of binary variables to take continuous values. In our example, we assume that in the first iteration only the 160 binary setup decision variables corresponding to the first 4 periods are restricted to be binary variables, and the remaining 320 binary variables are relaxed to be continuous. Because this relaxed subproblem has a much smaller number of binary variables, it is much easier to solve optimally. Once the binary solutions for the first 160 binary variables are determined, they are then fixed. In the second iteration, we then restrict the next 160 binary variables (corresponding to periods 5 through 8) to binary values, and only relax the remaining 160 binary variables (corresponding to periods 9 through 12) to be continuous. A good feasible solution can then be determined using this iterative approach.

To generate a better feasible solution, a variant of the above Relax-and-Fix strategy will be presented in this paper. We use the previous example to illustrate this variant of the Relax-and-Fix method. From our prior discussion, we know that in the first iteration, only the 160 binary setup variables corresponding to the first 4 periods are restricted to binary values, and these are fixed to the resulting binary values obtained in the first iteration. In the new strategy, suppose that we do not permanently fix all 160 binary variables, but fix only the 80 binary variables corresponding to the first two periods. In the second iteration, we then restrict the 160 binary variables corresponding to periods three through six to take binary values; after this iteration, the 80 binary variables corresponding to periods three and four are then fixed to their binary solution values. A final solution is achieved by using this approach iteratively. In the following, we will use a more formal way to describe this "rolling" heuristic solution method.

4.1. The Relax-and-Fix Algorithm

Before we describe the method formally, we first define two parameters, α and γ; α defines the number of periods for which binary variables are required to take binary values in the MILP solution of the restricted problem, and γ is the number of periods for which the binary variables have been fixed based on the binary solutions at previous iteration(s). In the above example, α equals 4 and γ equals 2. At each iteration of the algorithm, we need to solve a small subproblem in which a subset of binary variables is relaxed to be continuous. To present the subproblem formulation more clearly, we introduce three subsets of the setup decision variables. The set TF defines the index set of the binary setup variables, y_{it}, that have been permanently fixed after previous iteration(s); the set TI defines the index set of binary setup variables that must take binary values in the current iteration, and the set TR defines the index set of setup variables that may take continuous values between 0 and 1. The union of the sets TF, TI, and TR corresponds to the full set of setup decision variables. For those (i,t) pairs in the set TF, we let y_{it}^F denote the fixed binary value of y_{it} determined at a previous iteration. For simplicity, we only discuss the subproblem corresponding to the FL formulation in the following, which we refer to as FL-SP(TF,TI,TR).

FL-SP(TF,TI,TR):

$$\min \sum_{i=1}^{I}\sum_{t=1}^{T} sc_i \cdot y_{it} + \sum_{i=1}^{I}\sum_{t=1}^{T}\sum_{p=t}^{T} ec_i \cdot (p-t) \cdot u_{itp} \tag{25}$$

Subject to: (9),(10), (11), (12), (13),

$$u_{itp} \geq 0,\ s_{it} \geq 0,\ s_{i0} = 0, \quad \forall\, i = 1,\dots,I, t = 1,\dots,T, p = t,\dots,T, \tag{26}$$

$$\begin{array}{ll} y_{it} = y_{it}^F & (i,t) \in \text{TF}, \\ y_{it} \in \{0,1\} & (i,t) \in \text{TI}, \\ 0 \leq y_{it} \leq 1 & (i,t) \in \text{TR}. \end{array} \tag{27}$$

Using the α and γ values as defined earlier, we can define the relax-and-fix algorithm using the FL formulation for the subproblems as follows:

Next, we define the feasible region associated with each formulation of subproblem FL-SP(TF,TI,TR) and the corresponding decision problem as follows:

$$P_{FL-SP(TF,TI,TR)} = \{(x,y,s,u) | (9)-(13), (26), (27)\}$$
$$Z_{FL-SP(TF,TI,TR)} = \min\{(1) | (x,y,s,u) \in P_{FL-SP(TF,TI,TR)}\}$$

Algorithm 2: The Relax-and-Fix Algorithm

```
$iter = 1$ ;
repeat
    $TF = \emptyset$ ;
    for $i = 1, ..., I$ and $t = 1, ..., (iter - 1)\gamma$ do
        $TF \leftarrow (i, t)$ ;
        Fix $y_{it}$ to its value at the prior iteration ;
    $TI = \emptyset$ ;
    for $i = 1, ..., I$ and $t = (iter - 1)\gamma + 1, ..., (iter - 1)\gamma + \alpha$ do
        $TI \leftarrow (i, t)$ ;
    $TR = \emptyset$ ;
    for $i = 1, ..., I$ and $t = (iter - 1)\gamma + \alpha + 1, ..., T$ do
        $TR \leftarrow (i, t)$ ;
    Solve FL-SP(TF,TI,TR) and obtain corresponding $y_{it}$ values ;
    $iter = iter + 1$
until $TI = \emptyset$ ;
```

Using a similar approach, the additional models we have discussed (in addition to the FL problem formulation) can also be easily formulated as subproblems, and we refer to the corresponding decision subproblem for S-ILS and SR as $Z_{S-ILS-SP(TF,TI,TR)}$ and $Z_{SR-SP(TF,TI,TR)}$, respectively. We next provide an additional important property associated with the Relax-and-Fix approach.

Corollary 2. $Z_{FL-SP(TF,TI,TR)} = Z_{S-ILS-SP(TF,TI,TR)} = Z_{SR-SP(TF,TI,TR)}$. *Therefore, the Relax-and-Fix algorithm generates the same upper bound on the objective function value when it is implemented using S-ILS, FL, or SR.*

This follows from Corollary 1. At the first iteration of the Relax-and-Fix algorithm, we know that the subproblems implemented using these three formulations are equivalent. Therefore, the solutions that result in terms of the decision variables y_{it} will be equivalent, and the same values will be fixed no matter which formulation is used. Our results imply that the subproblems at subsequent iterations will also be equivalent, leading to the same final upper bound, regardless of formulation.

5. Computational Test Results

We have presented results that demonstrate the equivalence of strong formulations for the CLST-ML in Sections 3 and 4. However, the computational effort required for solving the problem is formulation dependent. In this section, we conduct computational experiments on test instances generated by Tempelmeier and Derstroff [1996] and Stadtler [2003] to show the differences in computational effort required for the Relax-and-Fix algorithm when using the S-ILS, FL, and SR formulations, respectively. Meanwhile, to gain a more general sense of the efficiency associated with these three formulations, computational tests were also performed on the linear programming relaxation using a commercial solver, CPLEX 11.2. Our experiments were performed on sets A+, B+, C, and D, where A+ contains 72 test instances, B+ contains 144 test instances, C contains 108 test instances, and D contains 96 test instances. Sets A+ and B+ involve problems with 10 items, 24 periods and 3 machines, while sets C and D involve problems with 40 items, 16 periods and 6 machines. There is no setup time for sets A+ and C, but sets B+ and D include positive setup times.

The data sets were constructed using a full factorial design with seven factors:

1. *Operations structure*: we consider two settings, one of which is a general multi-level structure, while the other is an assembly structure.
2. *Resource assignment*: there are two settings, acyclic and cyclic. In the acyclic problems, no item may use the same machine as one or more of its predecessors, though such situations are permitted in the cyclic problems. Acyclic problems are generally more difficult to solve than the corresponding cyclic problems, and we only consider acyclic problems.
3. *Setup time*: we consider three settings denoted by 0, 1, and 4, where 0 indicates that there is no setup time, and 1 and 4 indicate that setup times are required before producing an item; 1 indicates a slight, or small setup time and 4 indicates a value of high setup time.
4. *Coefficient of demand variation*: we consider two settings denoted by 1 and 2, where 1 indicates slight demand variation and 2 indicates sizable variation.
5. *Resource utilization*: we consider three settings denoted by 1, 2, and 3, where 1 represents high utilization, 2 represents medium utilization, and 3 corresponds to low utilization. The original data sets include five settings for resource utilization; we use only three of these settings here because the remaining two settings are quite similar to setting 3.
6. *TBO*: TBO denotes the time between orders, and we consider three settings denoted as 2, 3 and 4, where 2 indicates a high TBO, 3 indicates a medium TBO, and 4 corresponds to a low TBO.
7. *Amplitude of seasonal pattern*: we consider three settings denoted by 0, 1, and 2, where 0 indicates no seasonality for item demands, 1 indicates slight seasonality, and 2 indicates strong seasonality.

More detailed information about the experimental setup can be found in Stadtler [2003]. All of the test instances were run on a PC with an Intel Pentium 4 3.16 GHz processor. The model formulations and the heuristic algorithm were implemented using GAMS, a high-level algebraic modeling language. CPLEX 11.2 was used in the computational experiments to solve linear and mixed integer programming problems.

5.1. Computational Tests on the LP Relaxation

As previously mentioned, an efficient existing separation algorithm exists, which we implemented as a preprocessing procedure to generate cuts for the S-ILS formulation (see Algorithm 1). To gain the benefits of this algorithm, we have implemented it in the solution of S-ILS. The computational results listed in Table 1 demonstrate that the number of iterations needed for this algorithm is approximately 15, even for the difficult test instances within data set C. Given that it can significantly reduce the sizes of MIP lot sizing problems, spending a little extra computational effort using this algorithm is then recommended.

Table 1: Iteration number of separation algorithm for the full factorial experiment of datasets $A+$, $B+$, C and D.

	Utilization			TBO		Seasonality		Demand Var.	
Data Sets	U1	U2	U3	TBO3	TBO4	S1	S2	D1	D2
A+	10.29	12.00	15.67	12.14	13.17	12.63	12.88	12.33	12.97
B+	11.33	13.69	17.54	13.04	15.33	14.21	13.90	14.13	14.25
C	15.50	15.78	16.69	15.28	16.75	15.81	15.22	16.00	15.98
D	15.81	16.06	15.19	14.92	16.46	16.00	15.38	15.45	15.69

Table 2 shows the differences in computational effort required for solving the LP relaxation of the S-ILS, FL, and SR formulations, respectively. This table tends to indicate that, for data sets A+ and B+ of medium sizes, S-ILS is the most efficient formulation with respect to solving the linear programming relaxation, although for data sets C and D of large size, SR becomes the most efficient formulation. For example, SR requires the least time to obtain LP relaxation solutions for 80 out of 98 instances within data set D. Although the data listed in this table shows that SR is efficient, there are a few disadvantages associated with SR that we need to emphasize. When compared with the other two formulations, SR is much more sensitive to certain characteristics of lot sizing problem instances. That is, it can be fairly efficient for solving instances that are not tightly capacitated, but is not very efficient for solving tightly capacitated constrained lot sizing problems, especially for problems with high seasonality. Our empirical results indicate that SR is the most efficient formulation for obtaining LP relaxation solutions for almost all of the test instances with low capacity utilization (which constitute about one-third of the instances), but not for others. This can be attributed to the fact that the SR formulation has a zero integrality gap in the single item uncapacitated case when setup times are zero. Having looser capacities and smaller setup times makes the problem behave more like an uncapacitated problem. Also, SR contains a network structure, and this would also help to solve it quickly.

Table 2: Comparisons of linear programming relaxation for data sets A+, B+, C and D.

	Solution Time			Shortest Time Used By		
	S-ILS	FL	SR	S-ILS	FL	SR
A+	0.19	0.42	0.30	48	0	24
B+	0.20	0.35	0.25	73	2	69
C	1.83	1.66	1.18	24	14	70
D	1.32	0.80	0.63	2	14	80

5.2. Computational Tests for Relax-and-Fix

For data sets A+ and B+, the values of α and γ in the Relax-and-Fix algorithm were set to 4 and 2, respectively. For data sets C and D, their values were both set to 1. Generally, it would be preferable to set a comparatively small time limit in the MILP solver when solving subproblems in the Relax-and-Fix algorithm in order to reduce the computational time. However, our purpose in conducting our computational tests is to determine which formulation is more efficient in terms of computational effort. Therefore, we set a comparatively larger time limit of 40 minutes for solving subproblems for data sets C and D. With respect to data set C, we found that many of the subproblems could not be solved to optimality even within 40 minutes. This hinders our ability to effectively find the true difference in computational effort for the different formulations. Therefore, to better characterize the difference in computational effort, we used an additional stopping criterion for data set C. That is, we stop solving a subproblem when the duality gap reaches 1.5%. The duality gap is calculated as the difference between the best upper and lower bound solution values, divided by the best upper bound solution value. Comparisons using formulations S-ILS, FL, and SR are shown in Tables 3, 4, 5, and 6, where computational times are expressed in minutes. In these tables, U1, U2, and U3 correspond to the utilization profile values 1, 2, and 3, respectively, and S0, S1, and S2 correspond to the seasonality profile values 0, 1, and 2, respectively. Other abbreviations follow this approach as well.

Computational results for the data sets without setup times, A+ and C, are given in Tables 3 and 4, where the effects on computational times of the different factors are compared for each of the formulations, S-ILS, FL, and SR. According to the results shown in Table 3, we find that for data set A+, the S-ILS formulation is the most efficient, the FL formulation is the second most efficient, and the SR formulation is the least efficient. This is particularly true for the test instances with high capacity usage, where the computational time required for S-ILS is less than half of that required for SR. Except for the test instances with low capacity usage, we find that S-ILS is better than the other two formulations for the test instances with low or high TBO, low or high seasonality, and low or high demand variability. In terms of the average computational time for all test instances for data set A+, the computational time needed by S-ILS is about 97.6% that of FL and 66.6% that of SR for the general test instances, while the time required by S-ILS is about 77.3% that of FL and 57.6% that of SR for the assembly test instances. When it comes to data set C, which contains larger problem sizes, we observe similar results, i.e., the S-ILS formulation is better than the other formulations with the exception of test instances with low capacity usage, and the efficiency is more obvious for hard test instances that are highly capacitated.

Computational results for the data sets with positive setup times, B+ and D, are given in Tables 5 and 6. According to the results for data set B+, S-ILS still appears to be the best choice on average when using the Relax-and-Fix algorithm. However, the advantages of S-ILS over the other two formulations are not as obvious for test instances with setup times when compared to those for test instances without setup times. In terms of the results on data set D, the computational effort needed for the three formulations is comparable, and the SR formulation is slightly better than others in the majority of test instances.

The computational effort needed to solve a lot sizing problem is also dependent on the

Table 3: Comparisons of Relax-and-Fix for data set A+.

		General			Assembly		
		S-ILS-T	FL-T	SR-T	S-ILS-T	FL-T	SR-T
Utilization	U1 (high)	1.15	1.27	2.65	4.58	8.23	13.83
	U2 (medium)	1.63	1.70	1.98	5.70	5.86	5.72
	U3 (low)	0.95	0.84	0.96	1.94	1.78	1.83
TBO	TBO3 (medium)	1.82	1.94	2.80	5.52	7.14	10.36
	TBO4 (low)	0.66	0.60	0.92	2.66	3.44	3.83
Demand Var	D1 (low)	1.35	1.36	2.01	4.22	5.39	7.48
	D2 (high)	1.14	1.18	1.72	3.95	5.19	6.72
Seasonality	S0 (low)	1.34	1.26	1.88	3.32	3.52	3.76
	S1 (medium)	1.32	1.32	2.01	5.00	6.58	9.89
	S2 (high)	1.07	1.23	1.69	3.94	5.76	7.64
	Average	1.24	1.27	1.86	4.09	5.29	7.10

In the table, -T corresponds to the computational time associated with a formulation when the Relax-and-Fix algorithm is implemented.

Table 4: Comparisons of Relax-and-Fix for data set C.

		General			Assembly		
		S-ILS-T	FL-T	SR-T	S-ILS-T	FL-T	SR-T
Utilization	U1 (high)	10.38	14.34	23.28	11.37	15.94	24.43
	U2 (medium)	12.80	15.42	13.23	17.91	22.94	22.21
	U3 (low)	5.38	6.90	3.54	5.87	4.52	4.78
TBO	TBO2 (high)	15.55	21.68	21.09	22.18	27.01	29.61
	TBO3 (medium)	7.44	9.50	10.65	7.48	9.81	13.76
	TBO4 (low)	4.46	6.04	8.31	5.49	6.58	8.05
Demand Var	D1 (high)	11.69	16.06	19.25	15.05	18.37	24.15
	D2 (low)	6.61	8.38	7.45	8.38	10.56	10.13
Seasonality	S0 (low)	7.30	9.82	9.09	6.53	9.09	9.40
	S1 (medium)	11.95	13.44	13.54	14.08	18.24	22.43
	S2 (high)	9.31	13.95	17.42	14.54	16.07	19.59
	Average	9.52	12.22	13.35	11.72	14.47	17.14

characteristics of the problem itself. According to the results shown in the above four tables, it seems that the capacity usage has a remarkable influence on the computational time needed for solving problems by Relax-and-Fix. In the case of set D, the computational time needed for highly capacitated problems is almost ten times the computational time needed for problems that are not tightly capacitated problems. From our computational

Table 5: Comparisons of Relax-and-Fix for data set B+.

		General			Assembly		
		S-ILS-T	FL-T	SR-T	S-ILS-T	FL-T	SR-T
Setup Time	Setup1 (low)	1.74	2.04	2.88	4.21	4.85	5.81
	Setup4 (high)	1.58	1.63	2.03	3.07	3.14	3.29
Utilization	U1 (high)	2.08	2.64	4.07	5.55	6.55	8.39
	U2 (medium)	1.77	1.84	2.14	3.73	3.92	3.70
	U3 (low)	1.12	1.02	1.16	1.64	1.51	1.56
TBO	TBO3 (medium)	2.41	2.79	3.60	4.75	5.18	6.12
	TBO4 (low)	0.90	0.88	1.31	2.53	2.80	2.98
Demand Var	D1 (high)	1.79	1.91	2.59	3.21	3.54	4.14
	D2 (low)	1.53	1.76	2.33	4.07	4.45	4.95
Seasonality	S0 (low)	1.59	1.55	2.30	3.53	3.59	4.08
	S1 (medium)	1.68	1.84	2.44	1.68	1.84	2.44
	S2 (high)	1.71	2.12	2.63	3.47	4.08	5.05
	Average	1.66	1.83	2.46	3.64	3.99	4.55

Table 6: Comparisons of Relax-and-Fix for data set D.

		General			Assembly		
		S-ILS-T	FL-T	SR-T	S-ILS-T	FL-T	SR-T
Setup Time	Setup1 (low)	38.79	42.63	40.25	15.64	17.18	15.48
	Setup4 (high)	26.00	27.88	24.08	5.16	4.90	4.40
Utilization	U1 (high)	59.75	70.50	70.44	23.21	25.56	22.66
	U2 (medium)	26.63	26.50	19.88	6.52	6.03	5.86
	U3 (low)	10.81	8.75	6.19	1.46	1.53	1.30
TBO	TBO3 (medium)	39.25	43.25	37.29	12.94	13.54	12.65
	TBO4 (low)	25.54	27.25	27.04	7.85	8.54	7.23
Demand Var	D1 (high)	37.46	41.33	37.79	10.50	10.22	8.77
	D2 (low)	27.33	29.17	26.54	10.29	11.86	11.11
Seasonality	S1 (medium)	32.38	35.38	32.19	11.17	11.32	10.49
	S2 (high)	32.42	35.13	32.17	9.63	10.76	9.39
	Average	32.40	35.25	32.15	10.40	11.04	9.94

results, we observe that the reason S-ILS works comparatively better than the other two formulations (for the tightly capacitated problems) is because the solution procedure for S-ILS is able to generated substantially more nodes than the other two formulations. The advantage of the node-generation speed of S-ILS over the other two formulations is less noticeable in the case of medium and low capacity utilization problems, and therefore its performance is closer to the performance of the other two formulations.

The TBO and demand variance also have a significant effect on computational time. For example, the computational time needed for the test instances with high and medium values of TBO is approximately two to three times more than the test instances with low values of TBO. In contrast, the influence of seasonality on solution time is less noticeable.

Table 7: More comparisons of Relax-and-Fix for data set A+, B+, C and D.

	Solution Time			Shortest Time Used By		
	S-ILS	FL	SR	S-ILS	FL	SR
A+	2.66	3.28	4.48	43	19	10
B+	2.65	2.91	3.50	75	49	20
C	10.62	13.34	15.24	64	19	25
D	21.40	23.15	21.05	33	14	49

To validate whether the above results are exceptionally influenced by only a few problem instances, Table 7 shows the number of instances that are solved the most quickly by each formulation (S-ILS, FL, or SR) when the Relax-and-Fix heuristic is implemented. The information in this table shows that, on average, S-ILS outperforms the other two formulations.

5.3. Computational Tests Using CPLEX Directly

To gauge the performance of each of the three aforementioned formulations in an exact solution algorithm, we tested all 420 instances using an MIP solver, CPLEX 11.2, with default settings. For data sets A+ and B+, the limit of solution time was set to 15 minutes, while the limit was set to 30 minutes for data sets C and D due to their complexity.

Table 8: Comparisons of CPLEX for data sets A+, B+, C and D.

	Duality Gaps			Best Solutions Found By		
	S-ILS	FL	SR	S-ILS	FL	SR
A+	16.44%	16.90%	17.39%	37	18	17
B+	15.24%	15.80%	16.29%	75	40	29
C	13.34%	14.31%	14.25%	60	24	24
D	13.88%	15.03%	16.43%	32	22	23

* S-ILS, FL and SR obtained the same feasible solution for 19 test instances.

The computational results are given in Table 8, in which the duality gaps and the number of instances in which the best feasible solution was achieved for each formulation are listed. To provide a fair comparison of the feasible solutions associated with each formulation, the lower bound yielded by the LP relaxation was used for calculating the duality gaps, which were computed as (upper bounds - lower bounds)/lower bounds. According to the results shown in this table, on average, CPLEX achieves the smallest duality gaps for all test sets when it is implemented using the S-ILS formulation. Meanwhile, the number of instances in which the best feasible solution was obtained using the S-ILS formulation was 223 out of the 400 test instances, compared with 123 obtained by FL and 112 obtained by SR. Based on these results, we recommend that the preference order for using these three formulations be S-ILS, FL and SR when they are solved directly using CPLEX.

5.4. Complexities of the Formulations

In this subsection, we discuss the key attributes pertaining to these formulations in greater detail. The number of columns, rows, and non-zeros in a formulation provide an indication of its size, and also strongly influence how efficiently a problem can be solved using an MIP solver. To provide guidelines on the relative efficiency associated with the three formulations, we list their numbers of columns, rows, and non-zeros in Table 9. In this table, the numbers within parentheses indicate the remaining columns or rows for the corresponding formulation after running the solver's pre-solve procedure, while the numbers to the left of those in parentheses indicate the number of columns or rows before performing the pre-solve. According to the table, the size of the original S-ILS formulation is smaller than other two formulations. More specifically, this formulation has a smaller number of columns, rows, and non-zeros than FL with only one exception for data set D; it also has a much smaller number of columns and non-zeros than that of SR, even though its number of rows is larger. Though we cannot conclude that S-ILS must be more efficient than the other formulations due to this characteristic, this reflects the potential efficiency of S-ILS over the other two formulations in general, purely in terms of problem size.

Table 9: Comparisons of problem complexities for data sets A+, B+, C and D.

	Rows			Columns		
Sets	S-ILS	FL	SR	S-ILS	FL	SR
A+	1698 (1627)	3803 (3507)	970 (870)	750	3496	3298
B+	1944 (1868)	3803 (3510)	970 (878)	737	3502	3296
C	5768 (5486)	7497 (6657)	2600 (2082)	1878	6632	6130
D	6034 (5869)	7497 (6638)	2593 (1959)	1811	6632	6109
	Non-zeros			**NGSpeed**		
Sets	S-ILS	FL	SR	S-ILS	FL	SR
A+	27881 (7608)	25345 (21427)	104135 (99057)	13522	3609	4503
B+	30016 (9372)	25505 (21492)	104295 (101255)	11069	3580	4714
C	94731 (39329)	47805 (39928)	156145 (155396)	1691	796	979
D	98436 (57082)	48465 (40281)	156785 (148653)	638	711	1307

Another significant factor shown in Table 9 is the number of nodes generated (on average for all test instances) within 2 minutes in the branching process (referred to as NGSpeed), which indicates the speed of node-generation for each of the formulations. The table shows that the S-ILS formulation has the fastest rate (with the exception in data set D) when generating nodes using an MIP solver. This shows that S-ILS is more likely to achieve better solutions, a reflection of the results in the previous subsections.

6. Conclusion

In this paper, we presented theoretical results on the equivalence of three prominent strong formulations for the CLST-ML when the binary requirement is relaxed on a subset of variables (as these variables may take continuous values between 0 and 1). These equivalence results have significant implications for heuristic methods, as certain decomposition-based heuristics will provide the same upper bound on optimal solution value, regardless of which formulation is used. Because of these equivalence results, the computational effort associated with these formulations then becomes the key reason to determine which formulation is the best choice. We used the Relax-and-Fix method as an example heuristic and conducted a large number of computational experiments on the data sets A+, B+, C, and D from Stadtler [2003]. Meanwhile, an exact method was also tested using CPLEX to illustrate the efficiency associated with each formulation. According to our computational results, the S-ILS formulation is the most efficient among the three well-known and strong formulations from the literature, and this is especially true for test instances with high capacity utilization. The SR formulation appears to be the least efficient formulation for most of the test instances, even though we find that it is the most efficient formulation for instances with low capacity utilization. Because the harder instances with high capacity utilization are more likely to be found in practice, we recommend a preference sequence of S-ILS, FL, and SR when implementing a heuristic method that may employ any of these formulations. Future work along this line of research may focus on demonstrating the equivalence of strong formulations for other extensions of lot sizing problems, such as the CLST-ML in which backlogging, setup carryover, overtime, and/or shortages are permitted.

Appendix A. For simplicity, assuming that (κ) is the index of a set of constraints, we let $c.(\kappa)$ denote constraints (κ). We also let * denote the feasible solution for the corresponding variable, or variable vector.

Proof of Theorem 2. First, let $(u^*, y^*, s^*) \in P_{FL}^{SUB}$ be any feasible point from the FL feasible set. Now, define the following point for all $t = 1, ..., T, i = 1, ..., I$:

$$x_{it}^* = \sum_{p=t}^{T} u_{itp}^*; e_{it}^* = \sum_{q=1}^{t}\sum_{p=t+1}^{T} u_{iqp}^*; s_{it}^* = \sum_{q=1}^{t}\sum_{p=t+1}^{T} u_{iqp}^* + \sum_{j\in\eta_i} r_{ij} \sum_{q=1}^{t}\sum_{p=t+1}^{T} u_{jqp}^*.$$

As (u^*, y^*, s^*) is a feasible point from the FL feasible set, we have $\sum_{p=1}^{t} u_{ipt}^* = ed_{it}$.

We then have that the following equation is valid:

$$\sum_{p=1}^{t} u_{ipt}^* = ed_{it} = gd_{it} + \sum_{j\in\eta_i} r_{ij} \cdot ed_{jt} = gd_{it} + \sum_{j\in\eta_i} r_{ij} \sum_{p=1}^{t} u_{jpt}^*. \tag{28}$$

As both $\sum_{p=1}^{t} u^*_{ipt}$ and $\sum_{p=1}^{t} u^*_{jpt}$ can be expressed as follows,

$$\sum_{p=1}^{t} u^*_{ipt} = \sum_{p=1}^{t-1} u^*_{ipt} + u^*_{itt} = \sum_{p=1}^{t-1} u^*_{ipt} + \sum_{p=t}^{T} u^*_{itp} - \sum_{p=t+1}^{T} u^*_{itp}, \tag{29}$$

The equations (28) can be rewritten as

$$\sum_{p=t}^{T} u^*_{itp} = gd_{it} + \sum_{j\in\eta_i} r_{ij} \cdot \sum_{p=t}^{T} u^*_{jtp} + \sum_{q=t+1}^{T} u^*_{itq} - \sum_{p=1}^{t-1} u^*_{ipt} + \sum_{j\in\eta_i}\sum_{q=t+1}^{T} r_{ij}u^*_{jtq} - \sum_{j\in\eta_i}\sum_{p=1}^{t-1} r_{ij}u^*_{jpt}. \tag{30}$$

Since we know

$$s^*_{it} - s^*_{i(t-1)} = \sum_{q=t+1}^{T} u^*_{itq} - \sum_{p=1}^{t-1} u^*_{ipt} + \sum_{j\in\eta_i}\sum_{q=t+1}^{T} r_{ij}u^*_{jtq} - \sum_{j\in\eta_i}\sum_{p=1}^{t-1} r_{ij}u^*_{jpt}, \tag{31}$$

the equations (30) can be simplified to

$$\sum_{p=t}^{T} u^*_{itp} = gd_{it} + \sum_{j\in\eta_i} r_{ij} \cdot \sum_{p=t}^{T} u^*_{jtp} + s^*_{it} - s^*_{i(t-1)}.$$

Via the relationships between x^* and u^*, and projecting the above equalities onto the space of S-ILS, the following equalities can be obtained:

$$x^*_{it} + s^*_{i(t-1)} = gd_{it} + \sum_{j\in\eta_i} r_{ij} \cdot x^*_{jt} + s^*_{it}. \tag{32}$$

Note that if i is an end item, then r_{ij} is zero. The equalities (32) can be expressed as $x^*_{it} + s^*_{i(t-1)} = gd_{it} + s^*_{it}$. As a consequence, we have proven that (x^*, y^*, s^*, e^*) satisfies $c.(2) \cap c.(3)$.

Using the same logic, we can easily prove that (x^*, y^*, s^*, e^*) satisfies $c.(3) \cap c.(4) \cap c.(5) \cap c.(6)$ by projecting constraints (10), (12), (13), and (14) onto the space of S-ILS, accordingly. To show that (x^*, y^*, s^*, e^*) satisfies (15), first note that for any given $i = 1, ..., I, t = 1, ..., T, p = 1, ..., T$, $u^*_{itp} \leq ed_{ip}y^*_{it}$ holds. Then, picking any $\ell = 1, ..., T$ and any subset $S \subseteq [1, \ell]$, we can sum these inequalities and obtain:

$$\sum_{t\in S}\sum_{p=t}^{\ell} u^*_{itp} \leq \sum_{t\in S}\sum_{p=t}^{\ell} ed_{ip}y^*_{it}.$$

Then, since $u^* \geq 0$, the following is valid:

$$\sum_{t\in S}\sum_{p=t}^{\ell} u^*_{itp} \leq \sum_{t\in S}\sum_{p=t}^{\ell} ed_{ip}y^*_{it} + \sum_{\substack{t=1\\ t\notin S}}^{\ell}\sum_{p=\ell+1}^{T} u^*_{itp}.$$

Then, adding $\sum_{t\in S}\sum_{p=\ell+1}^{T} u^*_{itp}$ to both sides, we have

$$\sum_{t\in S}\sum_{p=t}^{T} u^*_{itp} \leq \sum_{t\in S}\sum_{p=t}^{\ell} ed_{ip}y^*_{it} + \sum_{t=1}^{\ell}\sum_{p=\ell+1}^{T} u^*_{itp}.$$

Now, using the previous transformations, we obtain:

$$\sum_{t\in S} x^*_{it} \le \sum_{t\in S}\sum_{p=t}^{\ell} ed_{ip} y^*_{it} + e^*_{i\ell}.$$

Therefore, $(x^*, y^*, s^*, e^*) \in P^{SUB}_{S-ILS}$. This implies $proj_{(x,y,s,e)} P^{SUB}_{FL} \subseteq P^{SUB}_{S-ILS}$.

Next, we provide two examples that prove $proj_{(x,y,s,u)} P^{SUB}_{S-ILS} \subsetneq P^{SUB}_{FL}$. Note that for any $(x^*, y^*, s^*, e^*) \in P^{SUB}_{S-ILS}$ such that $s^*_{iT} > 0$, it is easy to see that $\sum\limits_{t=1}^{T} x^*_{it} > \sum\limits_{t=1}^{T} ed_{it}$, which is not possible for P^{SUB}_{FL} due to the summation of constraints (9) over all $t = 1, ..., T$. More specifically, when constraints (9) are added together over $t = 1, .., T$, the summation of the left side of constraints (9) is equal to the total demand from period 1 to T. This means that s^*_{iT} can be only zero, and then there is a conflict against the assumption. Hence a point from S-ILS with $s^*_{iT} > 0$ cannot be projected on the FL.

For the case in which $s^*_{iT} = 0$ holds for all $i = 1, ..., I$, consider the following example.

Example: Consider a problem with only three periods and one item (hence the subscript i omitted) with demand of $(5, 4, 5)$. Then, $x = (8, 2, 4)$ and $y = (1, 0.25, 0.8)$ is a valid solution for S-ILS. However, this solution cannot be mapped to FL, since $u_{12} \le 3$ and hence $u_{22} \ge 1 > 0.25 \times 2$. Therefore, $proj_{(x,y,s,u)} P^{SUB}_{S-ILS} \subsetneq P^{SUB}_{FL}$.

Appendix B.

PROOF OF THEOREM 3. This theorem can be proven by showing that $proj_{x,y,s,w}(P^{SUB}_{FL}) \subseteq P^{SUB}_{SR}$ and $proj_{x,y,s,u}(P^{SUB}_{SR}) \subseteq P^{SUB}_{FL}$.

First, we show that $proj_{x,y,s,w}(P^{SUB}_{FL}) \subseteq P^{SUB}_{SR}$ by supposing that $(x^*, y^*, s^*, u^*) \in P^{SUB}_{FL}$. Given the relationships between u^* and w^*, we know the following equations are valid:

$$u^*_{ipt} = \sum_{q=t}^{T} w^*_{ipq} \cdot ed_{it} \;\forall\, i = 1, \dots, I,\; p = 1, \dots, T,\; t = p, \dots, T.$$

Given $(x^*, y^*, s^*, u^*) \in P^{SUB}_{FL}$, the constraints below, derived from (9), are valid.

$$\sum_{p=1}^{t} u^*_{ipt} = ed_{it} \;\forall\, i = 1, \dots, I,\; t = 1, \dots, T.$$

By projecting these constraints onto the space of SR, we have:

$$\sum_{p=1}^{t}\sum_{q=t}^{T} w^*_{ipq} \cdot ed_{it} = ed_{it} \;\forall\, i = 1, \dots, I,\; t = 1, \dots, T.$$

Given the above equations, for $t = 1$, we know that $\sum\limits_{p=1}^{T} w^*_{i1p} = 1, \;\forall\, i = 1, \dots, I$. For $t \in [2, T]$, if we take the equalities for $t = t$, and then subtract the equalities for $t = t - 1$ side by side, we can obtain the following valid equalities:

$$\sum_{q=1}^{t-1} w^*_{iq,t-1} = \sum_{q=t}^{T} w^*_{itq} \;\forall\, i = 1, \dots, I,\; t = 2, \dots, T.$$

As a consequence, we have shown that (x^*, y^*, s^*, w^*) satisfies $c.(16) \cap c.(17)$. Similarly, we know that the following constraints are valid from (10):

$$s^*_{i,t-1} + \sum_{p=t}^{T} u^*_{itp} = gd_{it} + \sum_{j \in \eta_i} \sum_{p=t}^{T} r_{ij} u^*_{jtp} + s^*_{it} \; \forall\, i = I^E + 1, \dots, I, t = 1, \dots, T.$$

For any t $(t = 1, \dots, T)$, if we add the equations in Constraints (10) side by side for all t from 1 to g, then the following equations can be obtained:

$$\sum_{p=1}^{T} u^*_{i1p} + \sum_{p=2}^{T} u^*_{i2p} + \dots + \sum_{p=t}^{T} u^*_{itp} = \sum_{q=1}^{t} gd_{iq} + \sum_{j \in \eta_i} r_{ij} (\sum_{p=1}^{T} u^*_{j1p} + \sum_{p=2}^{T} u^*_{j2p} + \dots + \sum_{p=t}^{T} u^*_{jtp})$$

After simplification, the above equations can be expressed concisely as follows:

$$\sum_{q=1}^{t} \sum_{p=q}^{T} u^*_{iqp} = \sum_{q=1}^{t} gd_{iq} + \sum_{j \in \eta_i} r_{ij} (\sum_{q=1}^{t} \sum_{p=q}^{T} u^*_{jqp}) + s^*_{it}, \; \forall\, i = I^E + 1, \dots, I, t = 1, \dots, T.$$

Via the relationships between u^* and w^*, we have $\sum\limits_{q=1}^{t} \sum\limits_{p=q}^{T} u^*_{iqp} = \sum\limits_{q=1}^{t} \sum\limits_{p=q}^{T} ed_{iqp} \cdot w^*_{iqp}$, and projecting the above inequalities onto the space of SR, the equalities below can be obtained.

$$\sum_{q=1}^{t} \sum_{p=q}^{T} ed_{iqp} \cdot w^*_{iqp} = \sum_{q=1}^{t} gd_{iq} + \sum_{j \in \eta_i} r_{ij} (\sum_{q=1}^{t} \sum_{p=q}^{T} ed_{iqp} \cdot w^*_{iqp}) + s^*_{it}, \quad \forall\, i = I^E + 1, \dots, I, t = 1, \dots, T.$$

Hence, we have shown that (x^*, y^*, s^*, w^*) satisfies $c.(18)$. Again, using the same logic, we can prove (x^*, y^*, s^*, w^*) satisfies $c.(19) \cap c.(21) \cap c.(22)$ by projecting constraints (12), (13), and (14) onto the space of SR, accordingly.

To prove (x^*, y^*, s^*, w^*) satisfies $c.(20)$, we use the following relationships between u^* and w^*:

$$\sum_{p=t}^{T} u^*_{itp} = \sum_{p=t}^{T} \sum_{q=p}^{T} w^*_{itq} \cdot ed_{ip} = \sum_{p=t}^{T} w^*_{itp} \cdot ed_{itp}, \; \forall\, i \in [1, I],\; t \in [1, T].$$

Depending on such relationships, if we project c.(12) onto the space of SR, then (x^*, y^*, s^*, w^*) satisfying $c.(20)$ is obviously true.

Consequently, we have proven that if $(x^*, y^*, s^*, u^*) \in P^{SUB}_{FL}$, then $(x^*, y^*, s^*, w^*) \in P^{SUB}_{SR}$ is guaranteed so that $proj_{x,y,s,w}(P^{SUB}_{FL}) \subseteq P^{SUB}_{SR}$.

Conversely, the above proof process is reversible. If we let $(x^*, y^*, s^*, w^*) \in P^{SUB}_{SR}$, and then project all constraints within P^{SUB}_{SR} onto the space of FL, we can show that $(x^*, y^*, s^*, u^*) \in P^{SUB}_{FL}$ and $proj_{x,y,s,u}(P^{SUB}_{SR}) \subseteq P^{SUB}_{FL}$. The proof is complete. $\square$

Acknowledgement. This research was supported in part by the National Science Foundation under grant CMMI-00646697, and by the Air Force Office of Scientific Research under grant FA9550-07-1-0390. The authors are also thankful to two anonymous referees for their suggestions improving the presentation of the paper.

References

[1] Absi N., Kedad-Sidhoum S., 2007. MIP-based heuristics for multi-item capacitated lot-sizing problem with setup times and shortage costs. *RAIRO - Operations Research* 41, 171-192.

[2] Akartunalı, K. 2007. Computational methods for big bucket production planning problems: feasible solutions and strong formulation. Ph.D. Thesis, *University of Wisconsin-Madison*, Dept of Industrial Engineering.

[3] Akartunalı, K., Miller, A.J., 2009. A heuristic approach for big bucket multi-level production planning problems. *European Journal of Operational Research* 193 (2), 396-411.

[4] Akartunalı K., Miller, A.J., 2007. A computational analysis of lower bounds for big bucket production planning problems. *Available at Optimization Online,* `http://www.optimization-online.org/DB_HTML/2007/05/1668.html`.

[5] Barany, I., Van Roy, T.J., Wolsey L.A., 1984. Strong formulations for multi-item capacitated lot-sizing. *Management Science* 30 (10), 1255-1261.

[6] Belvaux, G., Wolsey L.A., 2000. Bc-prod: A specialized branch-and-cut system for lot-sizing problems. *Management Science* 46 (5), 724-738.

[7] Billington, P., J. McClain, L. Thomas. 1983. Mathematical programming approaches to capacity-constrained MRP systems: Review, formulation and problem reduction. *Management Science* 29 (10), 1126-1141.

[8] Denizel M., Altekin F.T., Sural H., Stadtler H. 2008. Equivalence of the LP relaxations of two strong formulations for the capacitated lot-sizing problem with setup times. *OR Spectrum* 30 (4), 773-785.

[9] Eppen, G.D., Martin, R.K., 1987. Solving multi-item capacitated lot-sizing problems using variable redefinition. *Operations Research* 35 (6), 832-848.

[10] Florian, M., Lenstra, J.K., Rinnooy Kan, H.G., 1980. Deterministic production planning: Algorithms and complexity. *Management Science* 26 (7), 669-679.

[11] Krarup, J., Bilde, O., 1977. Plant location, set covering and economic lotsizes: An O(mn) algorithm for structured problems. *Optimierung bei Graphentheoretischen und Ganzzahligen Probleme*. BirkhauserVerlag, 155-180.

[12] Mercé, C., Fontan, G., 2003. MIP-based heuristics for capacitated lotsizing problems. *International Journal of Production Economics* 85(1) 97-111.

[13] Nemhauser, G.L., Wolsey, L.A., 1988. *Integer and combinatorial optimization*, John Wiley & Sons, Inc.

[14] Salomon, M., 1991. Deterministic lot sizing models for production planning, Springer, Inc.

[15] Sahling, F., Buschkühl, L., Tempelmeier, H., and Helber, S., 2009. Solving a multi-level capacitated lot sizing problem with multi-period setup carry-over via a fix-and-optimize heuristic, *Computers & Operations Research* 36 (9), 2546-2553.

[16] Stadtler, H., 1997. Reformulations of the shortest route model for dynamic multi-item multi-level capaciated lotsizing. *OR Spectrum* 19 (2), 87-96.

[17] Stadtler, H., 2003. Multilevel lot sizing with setup times and multiple constrained resources: Internally rolling schedules with lot-sizing windows. *Operations Research* 51 (3), 487-502.

[18] Tempelmeier H., Derstroff M., 1996. A Lagrangean-based heuristic for dynamic multilevel multiitem constrained lotsizing with setup times. *Management Science* 42(5), 738-757.

[19] Tempelmeier, H., Helber S., 1994. A heuristic for dynamic multi-item multi-level capacitated lotsizing for general product structures. *European Journal of Operational Research* 75 (2), 296-311.

[20] Wu, T., Shi, L., Duffie, N., 2010. An HNP-MP approach for the capacitated multi-Item lot sizing problem with setup times. *IEEE Transactions on Automation Science and Engineering* 7 (3), 500-511.